Replication Report: Zou(2022) Unwatched Pollution: The Effect of Intermittent Monitoring on Air Quality, AER

Wonjong Kim

2022-12-09

This paper is a replication report of Zou(2022), which shows the effect of intermittent monitoring of environmental standard on polluting activities. Among the results, this paper shows replication result of figure 2 in Zou(2022), which shows pollution gap between on-monitor day and off-monitor days.

Data Description

Main Data: Monitor Data

The main dataset that this paper mainly process is PM monitor characteristics come from EPA's Air Quality System (AQS) for the years 2001 to 2013. This contains the locations of air quality monitors under the county. Among the various pollutant, Zou(2022) only uses PM10 and PM2.5 to exploit Aerosol data from Satellite image that is explained below. This replication paper deals with this EPA dataset to practice how to manupulates dataset to fit other external datasets.

Grid data

Zou(2022) exploits grid-level data to use satellite data. Therefore, this paper needs to merge grid data to every dataset to use. Although Zou(2022) starts from creat shape file (.shp) by using ArcGIS, this paper simply imports provided shapefile.

Other Dataset

This paper treats most GIS-related data as external dataset, so that simply imports them from Zou(2020)'s data archive. Brief descriptions for datasets are below.

Aerosol Satellite Data This dataset is a measure of atmospheric particle pollution (hereby "aerosol"). Data came from the National aeronautics and Space Administration (NASA) Moderate Resolution Imagining Spectroradiometer (MODIS) algorithm.

Air quality Action Day data from AirNow This dataset contains historical real time air quality forecast. Because it has latitude and longitude variable, this paper assigns grid-level information for each location. For the practice, this external data is under the datawork process.

Global Historical Climatology Network (GHCN) This dataset is used to exploit temperature and precipitation data

Datasets for regression

The datasets described below are used for regression result

County PM2.5 data It shows county level monthly/daily average PM2.5 concentration.

North American Regional Reanalysis (NARR) It contains windspeed and direction data, which is used to create monthly average wind speed index that affects PM concentration

Data Management

EPA Monitor data

This paper mainly focuses on dealing with EPA monitor data. Data covers from 2001 to 2013. Therefore, I import 2001 data first and verify relevant variables, and iterate in through remaining time periods.

```
# PM monitor data
# ------------------------------------ Import first
# file, 2001
drct <- paste0(getwd(), "/Data/1_build")

epa.2001 <- read.csv(file = paste0(drct, "/epa/raw/annual_all_2001.csv"),
    sep = ",", quote = "\"", header = TRUE, colClasses = "character")
setDT(epa.2001)

head(epa.2001, 10)
str(epa.2001)
summary(epa.2001)

## Just use PM10 and PM2.5
for (PM in c("PM10", "PM25")) {
    Pname <- ifelse(PM == "PM10", "epa.2001.PM10", "epa.2001.PM25")
    Pcode <- ifelse(PM == "PM10", "81102", "88101")
    Pstan <- ifelse(PM == "PM10", "PM10 24-hour 2006", "PM25 24-hour 2006")

    assign(Pname, epa.2001[Parameter.Code == Pcode])
```

```
    ## Check 1 obs is state + county + site + poc +
    ## sample duration + eventtype + pollutantstandard
    ## sort
    setorder(get(Pname), State.Code, County.Code, Site.Num,
        POC, Sample.Duration, Event.Type, Pollutant.Standard)

    ## Keep hourdata(2006 standard)
    assign(Pname, get(Pname)[Pollutant.Standard == Pstan])

    ## Keep relevant variables

    selectvar <- c("State.Code", "County.Code", "Site.Num",
        "POC", "Year", "Completeness.Indicator", "Valid.Day.Count",
        "Required.Day.Count", "Null.Data.Count", "Arithmetic.Mean",
        "X99th.Percentile", "X98th.Percentile", "X95th.Percentile",
        "X90th.Percentile", "X75th.Percentile", "X50th.Percentile",
        "X10th.Percentile")
    assign(Pname, get(Pname)[, ..selectvar])

}

epa.master.PM10 <- epa.2001.PM10
epa.master.PM25 <- epa.2001.PM25

# Iterate for 2002-2013 (and append it)
for (i in 2002:2013) {
    epa.append <- read.csv(file = paste0(drct, "/epa/raw/annual_all_",
        i, ".csv"), sep = ",", quote = "\"", header = TRUE,
        colClasses = "character")
    setDT(epa.append)

    for (PM in c("PM10", "PM25")) {
        Pname <- ifelse(PM == "PM10", "epa.append.PM10",
            "epa.append.PM25")
        Pcode <- ifelse(PM == "PM10", "81102", "88101")
        Pstan <- ifelse(PM == "PM10", "PM10 24-hour 2006",
            "PM25 24-hour 2006")

        assign(Pname, epa.append[Parameter.Code == Pcode])

        ## Check 1 obs is state + county + site + poc
        ## + sample duration + eventtype +
        ## pollutantstandard sort
        setorder(get(Pname), State.Code, County.Code, Site.Num,
```

```
            POC, Sample.Duration, Event.Type, Pollutant.Standard)

        ## Keep hourdata(2006 standard)
        assign(Pname, get(Pname)[Pollutant.Standard == Pstan])

        ## Keep relevant variables

        selectvar <- c("State.Code", "County.Code", "Site.Num",
            "POC", "Year", "Completeness.Indicator", "Valid.Day.Count",
            "Required.Day.Count", "Null.Data.Count", "Arithmetic.Mean",
            "X99th.Percentile", "X98th.Percentile", "X95th.Percentile",
            "X90th.Percentile", "X75th.Percentile", "X50th.Percentile",
            "X10th.Percentile")
        assign(Pname, get(Pname)[, ..selectvar])

    }
    epa.master.PM10 <- bind_rows(epa.master.PM10, epa.append.PM10)
    epa.master.PM25 <- bind_rows(epa.master.PM25, epa.append.PM25)
}

# Convert some variables as numeric
for (a in c("POC", "Year", "Valid.Day.Count", "Required.Day.Count",
    "Null.Data.Count", "Arithmetic.Mean", "X99th.Percentile",
    "X98th.Percentile", "X95th.Percentile", "X90th.Percentile",
    "X75th.Percentile", "X50th.Percentile", "X10th.Percentile")) {
    epa.master.PM10 <- epa.master.PM10[, `:=`(a, as.numeric(get(a))),
        with = FALSE]
    epa.master.PM25 <- epa.master.PM25[, `:=`(a, as.numeric(get(a))),
        with = FALSE]

}
```

The crucial variable for this research is the data for monitor day, which means each monitor captures the pollution at certain day and the other days it does not. Also, even I didn't exploit in this paper, entry and exit of monitor is one of the important variation.

```
# Monitor entry-exit info
# ------------------------------------------------
for (PM in c("PM10", "PM25")) {
    Pname <- ifelse(PM == "PM10", "entext.PM10", "entext.PM25")
    Pcode <- ifelse(PM == "PM10", "81102", "88101")

    entext <- read.csv(file = paste0(drct, "/epa/raw/aqs_monitors.csv"),
        sep = ",", quote = "\"", header = TRUE, colClasses = "character")
```

```
    str(entext)

    setDT(entext)

    # Drop unused obs (In paper)
    assign(Pname, entext[State.Code != "40" & County.Code !=
        "71" & Site.Number != "604" & POC != " 3" & Monitor.Type !=
        "SPM"])
    setDT(get(Pname))
    ## only PM10 or PM2.5
    assign(Pname, get(Pname)[Parameter.Code == "81102"])

    ## Drop monitors in CC (Canada, Mexico)

    setorder(get(Pname), State.Code, County.Code, Site.Number,
        Parameter.Code, POC)

    # Keep relevant variables
    selectvar <- c("State.Code", "County.Code", "Site.Number",
        "POC", "First.Year.of.Data", "Last.Sample.Date")
    assign(Pname, get(Pname)[, ..selectvar])
    # Rename variables
    setnames(get(Pname), c("Site.Number"), c("Site.Num"))

}

# Convert some variables as numeric
for (a in c("POC", "First.Year.of.Data")) {
    entext.PM10 <- entext.PM10[, `:=`(a, as.numeric(get(a))),
        with = FALSE]
    entext.PM25 <- entext.PM25[, `:=`(a, as.numeric(get(a))),
        with = FALSE]
}

str(epa.master.PM10)
str(entext.PM10)

# Merge annual & entry exit data
# ------------------------------------------------

setkey(epa.master.PM10, State.Code, County.Code, Site.Num,
    POC)
setkey(entext.PM10, State.Code, County.Code, Site.Num, POC)
# Merge m:1 (left join)
```

```
epa.PM10 <- entext.PM10[epa.master.PM10]

setkey(epa.master.PM25, State.Code, County.Code, Site.Num,
    POC)
setkey(entext.PM25, State.Code, County.Code, Site.Num, POC)
# Merge m:1 (left join)
epa.PM25 <- entext.PM25[epa.master.PM10]

# Remove
rm(list = ls()[!ls() %in% c("drct", "epa.PM10", "epa.PM25")])

# Melt(Collapse monitor level profile to site level
# profile) ----------------------------
epa.full <- bind_rows(epa.PM10, epa.PM25)
setDT(epa.full)
# Keep complete monitors (As paper did)
epa.full <- epa.full[Completeness.Indicator != "N"]

str(epa.full)
# Monitor offdays indicators (by monitor) 1-in-k
# days(6, 3, 1 days.)
epa.full <- epa.full[Required.Day.Count %in% c(60, 61),
    `:=`(schd1.1in6d, 1)]
epa.full <- epa.full[!Required.Day.Count %in% c(60, 61),
    `:=`(schd1.1in6d, 0)]

epa.full <- epa.full[Required.Day.Count %in% c(121, 122),
    `:=`(schd1.1in3d, 1)]
epa.full <- epa.full[!Required.Day.Count %in% c(121, 122),
    `:=`(schd1.1in3d, 0)]

epa.full <- epa.full[Required.Day.Count %in% c(365, 366),
    `:=`(schd1.1in1d, 1)]
epa.full <- epa.full[!Required.Day.Count %in% c(365, 366),
    `:=`(schd1.1in1d, 0)]

# Melt site-year level (See how many monitors in the
# sites has offdays, portion.)
epa.col <- epa.full[, .(schd1.1in6d = mean(schd1.1in6d),
    schd1.1in3d = mean(schd1.1in3d), schd1.1in1d = mean(schd1.1in1d)),
    by = .(State.Code, County.Code, Site.Num, Year)]

## Two indicators Offdays in any site
epa.col <- epa.col[schd1.1in6d != 0, `:=`(any1in6d, 1)]
```

```
epa.col <- epa.col[schd1.1in6d == 0, `:=`(any1in6d, 0)]

epa.col <- epa.col[schd1.1in3d != 0, `:=`(any1in3d, 1)]
epa.col <- epa.col[schd1.1in3d == 0, `:=`(any1in3d, 0)]

epa.col <- epa.col[schd1.1in1d != 0, `:=`(any1in1d, 1)]
epa.col <- epa.col[schd1.1in1d == 0, `:=`(any1in1d, 0)]

## Offdays in all sites
epa.col <- epa.col[schd1.1in6d == 1, `:=`(all1in6d, 1)]
epa.col <- epa.col[schd1.1in6d != 1, `:=`(all1in6d, 0)]

epa.col <- epa.col[schd1.1in3d == 1, `:=`(all1in3d, 1)]
epa.col <- epa.col[schd1.1in3d != 1, `:=`(all1in3d, 0)]

epa.col <- epa.col[schd1.1in1d == 1, `:=`(all1in1d, 1)]
epa.col <- epa.col[schd1.1in1d != 1, `:=`(all1in1d, 0)]
```

Grid Data Import

EPA data are based on site level. Thus, I also import grid data to merge and use satellite grid level data.

```
### Grid data
GRD.raw <- sf::st_read(paste0(drct, "/epa/proc/gisout_site_to_grid_cw.shp"))
setDT(GRD.raw)

GRD <- GRD.raw[, c("statecode", "countycode", "sitenum",
    "OBJECTID")]
setnames(GRD, c("statecode", "countycode", "sitenum", "OBJECTID"),
    c("State.Code", "County.Code", "Site.Num", "gridID"))

setnames(GRD, c("State.Code", "County.Code", "Site.Num"),
    c("statecode", "countycode", "sitenum"))
```

Data management for regression and figure

EPA data is monitor level data. Yet all air pollution level (PM concentration) data are daily based data, so that I have to expand EPA monitor data from monitor level to daily-monitor level.

```
# Create Regression file: site main regression
# ------------------------------------
head(epa.col)
```

```
epa.col.id <- epa.col[, `:=`(siteyrID, 1:.N)]

## expand to daily panel
epa.daily <- epa.col
for (i in 1:365) {
    epa.daily <- bind_rows(epa.daily, epa.col)
}
setDT(epa.daily)
### Sort
setorder(epa.daily, State.Code, County.Code, Site.Num, Year)
### daily variable, day of year
epa.daily <- epa.daily[, `:=`(doy, 1:.N), by = .(State.Code,
    County.Code, Site.Num, Year)]
### generate date
epa.daily <- epa.daily[, `:=`(dt, paste0(1, "/", 1, "/",
    Year))]
epa.daily <- epa.daily[, `:=`(dt, as.Date(dt, "%m/%d/%Y") +
    doy - 1)]

### Date info
epa.daily <- epa.daily[, `:=`(month, month(dt))]
epa.daily <- epa.daily[, `:=`(dow, wday(dt))]

## Generate countyfip code
epa.daily <- epa.daily[, `:=`(countyfip, paste0(State.Code,
    County.Code))]
epa.daily <- epa.daily[, `:=`(countyfip, as.numeric(countyfip))]
```

Also, external data that I explained above are needed for data analysis. The steps below are merging process to use a data by county level and grid level.

```
# Merge relevant data
# ------------------------------------------------------

## merge with GHCN data (weather)
setnames(epa.daily, c("State.Code", "County.Code", "Site.Num",
    "Year"), c("statecode", "countycode", "sitenum", "year"))

setkey(epa.daily, countyfip, year, doy)
setkey(GHCN, countyfip, year, doy)

master.dat <- epa.daily[GHCN]
master.dat <- master.dat[!statecode %in% NA]  # keep matched data

## merge with grid data
```

```
master.dat <- master.dat[, `:=`(statecode, as.numeric(statecode))]
master.dat <- master.dat[, `:=`(countycode, as.numeric(countycode))]
master.dat <- master.dat[, `:=`(sitenum, as.numeric(sitenum))]
setkey(master.dat, statecode, countycode, sitenum)
setkey(GRD, statecode, countycode, sitenum)

master.dat <- master.dat[GRD]
master.dat <- master.dat[!year %in% NA]  # keep matched data

## merge airosol data
setkey(master.dat, gridID, year, doy)
setkey(modis.raw, gridID, year, doy)

master.dat <- modis.raw[master.dat]
master.dat <- master.dat[!aod %in% NA]  # keep matched data

## merge observed PM2.5 data, external data
pm25 <- haven::read_dta(paste0(drct, "/misc/pm25_site_day.dta"))
setDT(pm25)

head(pm25)

setkey(master.dat, statecode, countycode, sitenum, year,
    doy)
setkey(pm25, statecode, countycode, sitenum, year, doy)

master.dat <- pm25[master.dat]
master.dat <- master.dat[!pm25_inc %in% NA]  # keep matched data

master.dat.temp <- master.dat
```

One more process is needed for regression. This research is focus on the effect of monitoring off-day. Zou(2022) want to show whether the air pollution level in off-days are significantly higher than on-day pollution level. Therefore, it creates the variable that indicate the cycle of monitor on-day.

```
# Generate cyclical event day
# ------------------------------------------------
Cyclic <- master.dat[, c("siteyrID", "statecode", "countycode",
    "sitenum", "year", "doy")]
Cyclic <- Cyclic[, `:=`(cycle_ID, 1:.N)]

Cyclic <- Cyclic[, `:=`(eday1, doy + 1 - 1)]
Cyclic <- Cyclic[, `:=`(eday2, doy + 2 - 1)]
Cyclic <- Cyclic[, `:=`(eday3, doy + 3 - 1)]
```

```
Cyclic <- Cyclic[, `:=`(eday4, doy + 4 - 1)]
Cyclic <- Cyclic[, `:=`(eday5, doy + 5 - 1)]
Cyclic <- Cyclic[, `:=`(eday6, doy + 6 - 1)]

Cyc.long <- melt(Cyclic, id.vars = c("cycle_ID", "siteyrID",
    "year"), measure.vars = patterns("^eday"), variable.name = "eday",
    value.name = "doy")
Cyc.long <- Cyc.long[, `:=`(eday, str_remove(eday, "eday"))]
Cyc.long <- Cyc.long[, `:=`(eday, as.numeric(eday))]

## merge with master data
setkey(master.dat.temp, siteyrID, year, doy)
setkey(Cyc.long, siteyrID, year, doy)

master.dat <- Cyc.long[master.dat.temp]

## 1 in 3 day cycle
master.dat <- master.dat[, `:=`(eday_1in3, eday)]
master.dat <- master.dat[eday_1in3 == 4, `:=`(eday_1in3,
    eday - 3)]
master.dat <- master.dat[eday_1in3 == 5, `:=`(eday_1in3,
    eday - 3)]
master.dat <- master.dat[eday_1in3 == 6, `:=`(eday_1in3,
    eday - 3)]

## variables for regression
master.dat <- master.dat[, `:=`(tmean, (tmax + tmin)/2)]
master.dat <- master.dat[!tmean %in% NA]
master.dat <- master.dat[, `:=`(g10_tmean, min(max(10 *
    floor(tmean/10), 10), 90))]

## Use wind speed data
narr <- haven::read_dta(paste0(drct, "/narr/narr_county_day.dta"))
setDT(narr)

setkey(master.dat, countyfip, year, doy)
setkey(narr, countyfip, year, doy)

master.dat <- master.dat[narr]
master.dat <- master.dat[!cycle_ID %in% NA]  # keep matched data

master.dat <- master.dat[, `:=`(g_wdsp, cut(wdsp, 5, labels = FALSE))]
setorder(master.dat, cycle_ID, eday)
```

```
master.dat <- master.dat[, `:=`(g_wdsp, g_wdsp[1]), by = cycle_ID]

## Log aod
master.dat <- master.dat[, `:=`(lnaod, log(aod))]
master.dat <- master.dat[!lnaod %in% NA]
master.dat <- master.dat[!lnaod %in% -Inf]

## group site
master.dat <- master.dat[, `:=`(g_site, .GRP), by = .(statecode,
    countycode, sitenum)]


## 1in6 treat
master.dat <- master.dat[, `:=`(offday, 1)]
master.dat <- master.dat[eday != 1, `:=`(offday, 0)]

## 1in3 treat
master.dat <- master.dat[, `:=`(offday_1in3, 1)]
master.dat <- master.dat[eday_1in3 != 1, `:=`(offday_1in3,
    0)]

## Scale AOD
master.dat <- master.dat[, `:=`(aod, aod * 100)]
save(master.dat, file = paste0(getwd(), "/Data_local/Master.Rda"))
```

Replication Result

This paper replicate the figure 2 in Zou(2022), which shows the significant gap of air quality within off-day of monitors. It shows the average air pollution level before 3 days of scheduled monitor and after 2 days of scheduled monitor. By regression with dummy variable to derive average and standard error of each day's variable, figures shows that air pollution levels in off-day are statistically higher than scheduled day to monitor the pollution level.

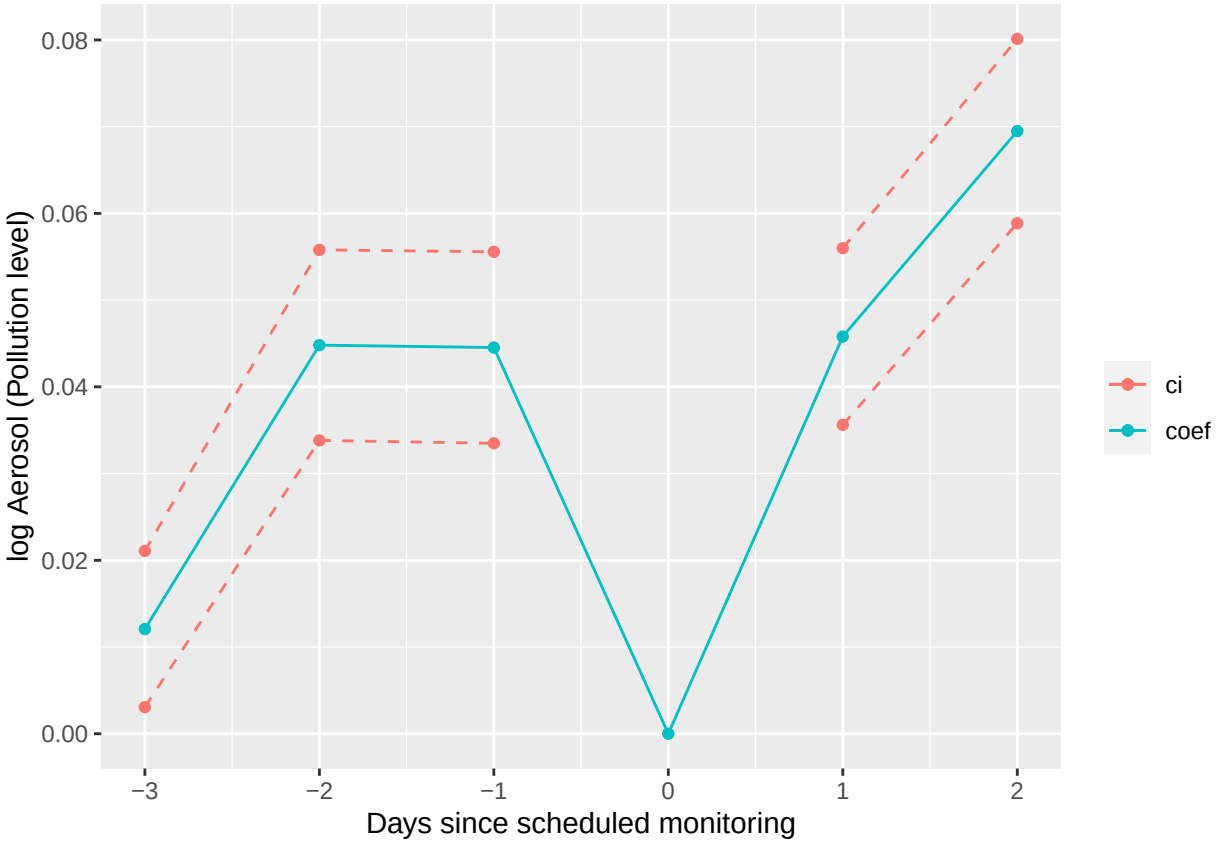
```
load(file = paste0(getwd(), "/Data_local/Master.Rda"))
reg <- lm(lnaod ~ factor(eday), na.action = na.omit, data = master.dat)
# print(reg) summary(reg)

stargazer::stargazer(reg, type = "text")
```

```
##
## ================================================
##                          Dependent variable:
##                      ---------------------------
##                                 lnaod
## ------------------------------------------------
```

```
## factor(eday)2                0.046***
##                               (0.005)
##
## factor(eday)3                0.069***
##                               (0.005)
##
## factor(eday)4                0.012***
##                               (0.005)
##
## factor(eday)5                0.045***
##                               (0.006)
##
## factor(eday)6                0.045***
##                               (0.006)
##
## Constant                     -2.074***
##                               (0.002)
##
## ------------------------------------------------
## Observations                   391,376
## R2                              0.001
## Adjusted R2                     0.001
## Residual Std. Error      0.974 (df = 391370)
## F Statistic         50.988*** (df = 5; 391370)
## ================================================
## Note:                *p<0.1; **p<0.05; ***p<0.01
```

```
cis <- coefci(reg)

Result <- as.data.frame(cis)
setDT(Result)

Result <- bind_cols(Result, as.data.frame(reg$coefficients))
setDT(Result)

Result <- Result[, `:=`(eday, 1:.N)]
setnames(Result, c(1, 2, 3, 4), c("ci_lower", "ci_upper",
    "coef", "eday"))

Result <- Result[eday == 1, `:=`(ci_lower, NA)]
Result <- Result[eday == 1, `:=`(ci_upper, NA)]
Result <- Result[eday == 1, `:=`(coef, 0)]

Result <- Result[, `:=`(eday, eday - 7)]
Result <- Result[eday < -3, `:=`(eday, eday + 6)]
```

```
setorder(Result, eday)

ggplot(Result, aes(x = eday)) + geom_line(aes(y = coef,
    color = "coef")) + geom_line(aes(y = ci_lower, color = "ci"),
    linetype = "dashed") + geom_line(aes(y = ci_upper, color = "ci"),
    linetype = "dashed") + geom_point(aes(y = coef, color = "coef")) +
    geom_point(aes(y = ci_lower, color = "ci")) + geom_point(aes(y = ci_upper,
    color = "ci")) + labs(x = "Days since scheduled monitoring",
    y = "log Aerosol (Pollution level)", colour = " ")
```



Lastly, I tried to replicate part of result from table 2, which shows the effect of offdays on air pollution. This replication only shows 1 in 6 offdays and 1 in 3 offdays. And odd column indicates sites with any 1 in 6 day monitor, and even column indicates any 1 in 3 day monitors. Due to the lack of the process, it shows offdays of monitors even reduce the air pollution.

```
load(file = paste0(getwd(), "/Data_local/Master.Rda"))

reg1 <- lm(lnaod ~ offday, subset = (any1in6d == 1), na.action = na.omit,
    data = master.dat)
reg2 <- lm(lnaod ~ offday, subset = (any1in6d == 0), na.action = na.omit,
```

```
    data = master.dat)
reg3 <- lm(lnaod ~ offday_1in3, subset = (any1in6d == 1),
    na.action = na.omit, data = master.dat)
reg4 <- lm(lnaod ~ offday_1in3, subset = (any1in6d == 0),
    na.action = na.omit, data = master.dat)

stargazer::stargazer(reg1, reg2, reg3, reg4, title = "Regression Estimates of the Pollut
    type = "latex", single.row = FALSE, column.sep.width = "-12pt",
    font.size = "scriptsize", header = FALSE)
```

Table 1: Regression Estimates of the Pollution Gap: Replication

	Dependent variable:			
	lnaod			
	(1)	(2)	(3)	(4)
offday	−0.082***	−0.013***		
	(0.004)	(0.005)		
offday_1in3			−0.097***	−0.017***
			(0.004)	(0.005)
Constant	−1.946***	−2.139***	−1.923***	−2.134***
	(0.003)	(0.003)	(0.004)	(0.003)
Observations	227,449	163,927	227,449	163,927
R^2	0.002	0.00005	0.002	0.0001
Adjusted R^2	0.002	0.00004	0.002	0.0001
Residual Std. Error	1.008 (df = 227447)	0.917 (df = 163925)	1.008 (df = 227447)	0.917 (df = 163925)
F Statistic	376.526*** (df = 1; 227447)	8.075*** (df = 1; 163925)	491.053*** (df = 1; 227447)	14.392*** (df = 1; 163925)
Note:				*p<0.1; **p<0.05; ***p<0.01